SPYWOLF

Security Audit Report

Audit prepared for

Chain Factory

Completed on

February 03, 2024

OVERVIEW

This goal of this report is to review the main aspects of the project to help investors make an informative decision during their research process.

You will find a a summarized review of the following key points:

- ✔ Contract's source code
- ✔ Owners' wallets
- ✔ Tokenomics
- ✔ Team transparency and goals
- ✔ Website's age, code, security and UX
- ✔ Whitepaper and roadmap
- ✔ Social media & online presence

The results of this audit are purely based on the team’s evaluation and does not guarantee nor reflect the projects outcome and goal

- SPYWOLF Team -

TABLE OF CONTENTS

CHAIN FACTORY

PROJECT DESCRIPTION

"With ChainFactory, users can choose from a variety of customizable templates and features, making it simple to create contracts tailored to your specific needs. It is designed to be user-friendly and intuitive, guiding users through the entire process step-by-step, providing a centralized platform to create, deploy, and manage your Smart-Contracts with ease."

Release Date: TBD

Category: Ecosystem

MAIN TOKEN CONTRACT

Token Name
rARB

Symbol
rARB

Contract Address
0xC4cD6b4504dE6Bfc1Fe0640dd68B7955235f5Eb6

Network
Ethereum Sepolia TESTNET

Language
Solidity

Deployment Date
Feb 01, 2024

Contract Type
Token with taxes

Total Supply
1,000,000

Status
Launched

TAXES

Buy Tax
5%

Sell Tax
5%

*Taxes can be changed in future

Our Contract Review Process

The contract review process pays special attention to the following:

- Testing the smart contracts against both common and uncommon vulnerabilities
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Blockchain security tools used:

- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat

TOKEN TRANSFERS STATS

Transfer Count	TESTNET
Uniq Senders	TESTNET
Uniq Receivers	TESTNET
Total Amount	TESTNET
Median Transfer Amount	TESTNET
Average Transfer Amount	TESTNET
First transfer date	TESTNET
Last transfer date	TESTNET
Days token transferred	TESTNET

SMART CONTRACT STATS

Calls Count	TESTNET
External calls	TESTNET
Internal calls	TESTNET
Transactions count	TESTNET
Uniq Callers	TESTNET
Days contract called	TESTNET
Last transaction time	TESTNET
Created	TESTNET
Create TX	TESTNET
Creator	TESTNET

FEATURED WALLETS

Owner address	0xBA799d418D1356ff5d225096d08951a3b45b6e4A
Marketing fee receivers	0xBA799d418D1356ff5d225096d08951a3b45b6e4A
LP address	TESTNET

TOP 3 UNLOCKED WALLETS

N/A	TESTNET
N/A	TESTNET
N/A	TESTNET

VULNERABILITY ANALYSIS

ID	Title	
SWC-100	Function Default Visibility	**Passed**
SWC-101	Integer Overflow and Underflow	**Passed**
SWC-102	Outdated Compiler Version	**Passed**
SWC-103	Floating Pragma	**Passed**
SWC-104	Unchecked Call Return Value	**Passed**
SWC-105	Unprotected Ether Withdrawal	**Passed**
SWC-106	Unprotected SELFDESTRUCT Instruction	**Passed**
SWC-107	Reentrancy	**Passed**
SWC-108	State Variable Default Visibility	**Passed**
SWC-109	Uninitialized Storage Pointer	**Passed**
SWC-110	Assert Violation	**Passed**
SWC-111	Use of Deprecated Solidity Functions	**Passed**
SWC-112	Delegatecall to Untrusted Callee	**Passed**
SWC-113	DoS with Failed Call	**Passed**
SWC-114	Transaction Order Dependence	**Passed**
SWC-115	Authorization through tx.origin	**Passed**
SWC-116	Block values as a proxy for time	**Passed**
SWC-117	Signature Malleability	**Passed**
SWC-118	Incorrect Constructor Name	**Passed**

VULNERABILITY ANALYSIS

ID	Title	
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed
SWC-134	Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed

MANUAL
CODE REVIEW

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time.

We categorize these vulnerabilities by 4 different threat levels.

THREAT LEVELS

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance, functionality and should be fixed before moving to a live environment.

Low Risk

Issues on this level are minor details and warning that can remain unfixed.

Informational

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.

FOUND THREATS

⚠ Medium Risk

Owner can change reflections token.
If **router** (router has pair address) contract is set as reflection token, contract will halt on sell once it reaches the autoswap amount.

```
function setDividendReflection(address token) external onlyOwner {
    require(!_renounced.DEXRouterV2);

    if (address(_dividendToken) != token && _amountSwappedForDividendDistribution > 0) { revert("Unclaimed taxes"); }

    _setDividendReflection(token);
}

function _setDividendReflection(address token) internal {
    require((!_initialized && token == address(0)) || token == address(this) || token == _dex.WETH
    || IDEXFactoryV2(IDEXRouterV2(_dex.router).factory()).getPair(_dex.WETH, token) != address(0), "No Pair");

    _dividendToken = IERC20(token == address(0) ? address(this) : token);
}


function _transfer(address from, address to, uint256 amount) internal virtual override {
.....................
if (!_distributing && !_swapping && (from != _dex.pair && from != _dex.router)) {
  _autoSwap(false);
  _autoTaxDistribute();
}
.....................
}

function _autoSwap(bool force) internal lockSwapping {
.....................

 if (address(_dividendToken) != address(this) && address(_dividendToken) != _dex.WETH) {
    amountForDividendDistributionToSwap = _amountForDividendDistribution;

    if (!force && amountForDividendDistributionToSwap > _maxAutoSwapAmount) { amountForDividendDistributionToSwap = _maxAutoSwapAmount; }

    if ((force || amountForDividendDistributionToSwap >= _minAutoSwapAmount) && _balance[address(this)] >= amountForDividendDistributionToSwap) {
      uint256 reflectionAmount = _swapTokensForTokens(_dividendToken, amountForDividendDistributionToSwap);

      if (reflectionAmount > 0) {
        _tokensForDividendDistribution += reflectionAmount;
        _amountSwappedForDividendDistribution += amountForDividendDistributionToSwap;
        _amountForDividendDistribution -= amountForDividendDistributionToSwap;
      }
    }
  }
.....................
}

function _swapTokensForTokens(IERC20 token, uint256 amount) private returns (uint256 tokenAmount) {
uint256 tokenBalance = token.balanceOf(address(this));  // Halt because of absent of the present function
.....................
}
```

- Recommendation:
  - Only token contracts that can be traded in DEX should be set as reflection token.

FOUND THREATS

Informational

Owner can set buy/sell/transfer taxes up to 25%, until the functionality is renounced.
Owner can set penalty taxes up to 100%. Penalty taxes are applied only for certain period of time for sells up to 10 minutes after the initial launch.
Combined buy+sell = 25%.

```
function setTaxBeneficiary(uint8 slot, address account, uint24[3] memory percent, uint24[3] memory penalty) external onlyOwner {
    require(!_renounced.Taxable);

    _setTaxBeneficiary(slot, account, percent, penalty);
}

function _setTaxBeneficiary(uint8 slot, address account, uint24[3] memory percent, uint24[3] memory penalty) internal {
    require(slot < 5);
    require(account != address(this));

    taxBeneficiary storage _taxBeneficiary = _taxBeneficiary[slot];

    if (account == address(0xdEaD) && _taxBeneficiary.unclaimed > 0) { revert("Unclaimed taxes"); }

    unchecked {
      _totalTxTax += percent[0] - _taxBeneficiary.percent[0];
      _totalBuyTax += percent[1] - _taxBeneficiary.percent[1];
      _totalSellTax += percent[2] - _taxBeneficiary.percent[2];
      _totalPenaltyTxTax += penalty[0] - _taxBeneficiary.penalty[0];
      _totalPenaltyBuyTax += penalty[1] - _taxBeneficiary.penalty[1];
      _totalPenaltySellTax += penalty[2] - _taxBeneficiary.penalty[2];

      require(_totalTxTax <= 25 * _denominator && ((_totalBuyTax <= 25 * _denominator && _totalSellTax <= 25 * _denominator)
      && (_totalBuyTax + _totalSellTax <= 25 * _denominator)), "High Tax");
      require(_totalPenaltyTxTax <= 100 * _denominator
      && _totalPenaltyBuyTax <= 100 * _denominator && _totalPenaltySellTax <= 100 * _denominator, "Invalid Penalty");
    }

    _taxBeneficiary.account = account;
    _taxBeneficiary.percent = percent;
    _taxBeneficiary.penalty = penalty;

    if (!_taxBeneficiary.exists) { _taxBeneficiary.exists = true; }

    emit SetTaxBeneficiary(slot, account, percent, penalty);
}
```

FOUND THREATS

Informational

Owner can withdraw any tokens from the contract.
When this function is present, in cases tokens are sent into the contract by mistake or purposefully, contract's owner can retrieve them.

```
function recoverERC20(address token, address to, uint256 amount) external onlyOwner {
unchecked {
  uint256 balance = IERC20(token).balanceOf(address(this));
  uint256 allocated = token == address(this)
  ? _amountForTaxDistribution + _amountForDividendDistribution + _amountForLiquidity
  : (address(_taxToken) == token ? _tokensForTaxDistribution : 0)
  + (address(_dividendToken) == token ? _tokensForDividendDistribution : 0);

  require(balance - (allocated >= balance ? balance : allocated) >= amount, "Exceeds balance");
}

IERC20(token).transfer(to, amount);
}

function recoverNative(address payable to, uint256 amount) external onlyOwner {
unchecked {
  uint256 balance = address(this).balance;
  uint256 allocated = (address(_taxToken) == _dex.WETH ? _ethForTaxDistribution : 0)
  + (address(_dividendToken) == _dex.WETH ? _ethForDividendDistribution : 0);

  require(balance - (allocated >= balance ? balance : allocated) >= amount, "Exceeds balance");
}

(bool success, ) = to.call{ value: amount }("");

require(success);
}
```

FOUND THREATS

Informational

Owner can set max balance (max wallet) that address can hold as low as 0.1% of total supply, until the functionality is renounced.

```
function setMaxBalancePercent(uint24 percent) external onlyOwner {
    require(!_renounced.MaxBalance);

    unchecked {
      require(percent == 0 || (percent >= 100 && percent <= 100 * _denominator));
    }

    _setMaxBalancePercent(percent);

    emit SetMaxBalancePercent(percent);
}

function _setMaxBalancePercent(uint24 percent) internal {
    _maxBalancePercent = percent;
    _maxBalanceAmount = percent > 0 ? _percentage(_totalSupply, uint256(percent)) : 0;

    if (!_initialized) { emit SetMaxBalancePercent(percent); }
}

function _percentage(uint256 amount, uint256 bps) internal pure returns (uint256) {
    unchecked {
      return (amount * bps) / (100 * uint256(_denominator));
    }
}
```

FOUND THREATS

Informational

Owner can set penalty time for sells up to 10 minutes from trading start timestamp.
If holders sell their tokens before the time set, they will be subject to penalty taxes.

```
function setEarlyPenaltyTime(uint32 time) external onlyOwner {
    require(!_renounced.Taxable);
    require(time <= 600);

    _setEarlyPenaltyTime(time);
}

function _setEarlyPenaltyTime(uint32 time) internal {
    _earlyPenaltyTime = time;

    emit SetEarlyPenaltyTime(time);
}
```

FOUND THREATS

Informational

Users can gain token spend approval from address if they have the required signature.

```
function permit(address owner, address spender, uint256 value, uint256 deadline, uint8 v, bytes32 r, bytes32 s) external {
require(deadline >= block.timestamp, "Expired signature");

unchecked {
  bytes32 digest = keccak256(abi.encodePacked(hex"1901",
  _domainSeparator, keccak256(abi.encode(PERMIT_TYPEHASH, owner, spender, value, _nonces[owner]++, deadline))));
  address recoveredAddress = ecrecover(digest, v, r, s);

  require(recoveredAddress != address(0) && recoveredAddress == owner, "Invalid signature");
}

_approve(owner, spender, value);
}
```

- Recommendation:
  - Users should be cautious when signing messages on web3 dapps to prevent unauthorized approvals with their signature.

FOUND THREATS

Informational

Owner can enable trading once.
Once enabled trading cannot be disabled again.
Trading is initially disabled.

```
function enableTrading() external onlyOwner {
    require(!_renounced.DEXRouterV2);
    require(_tradingEnabled == 0, "Already enabled");

    _tradingEnabled = _timestamp();
}
```

FOUND THREATS

Informational

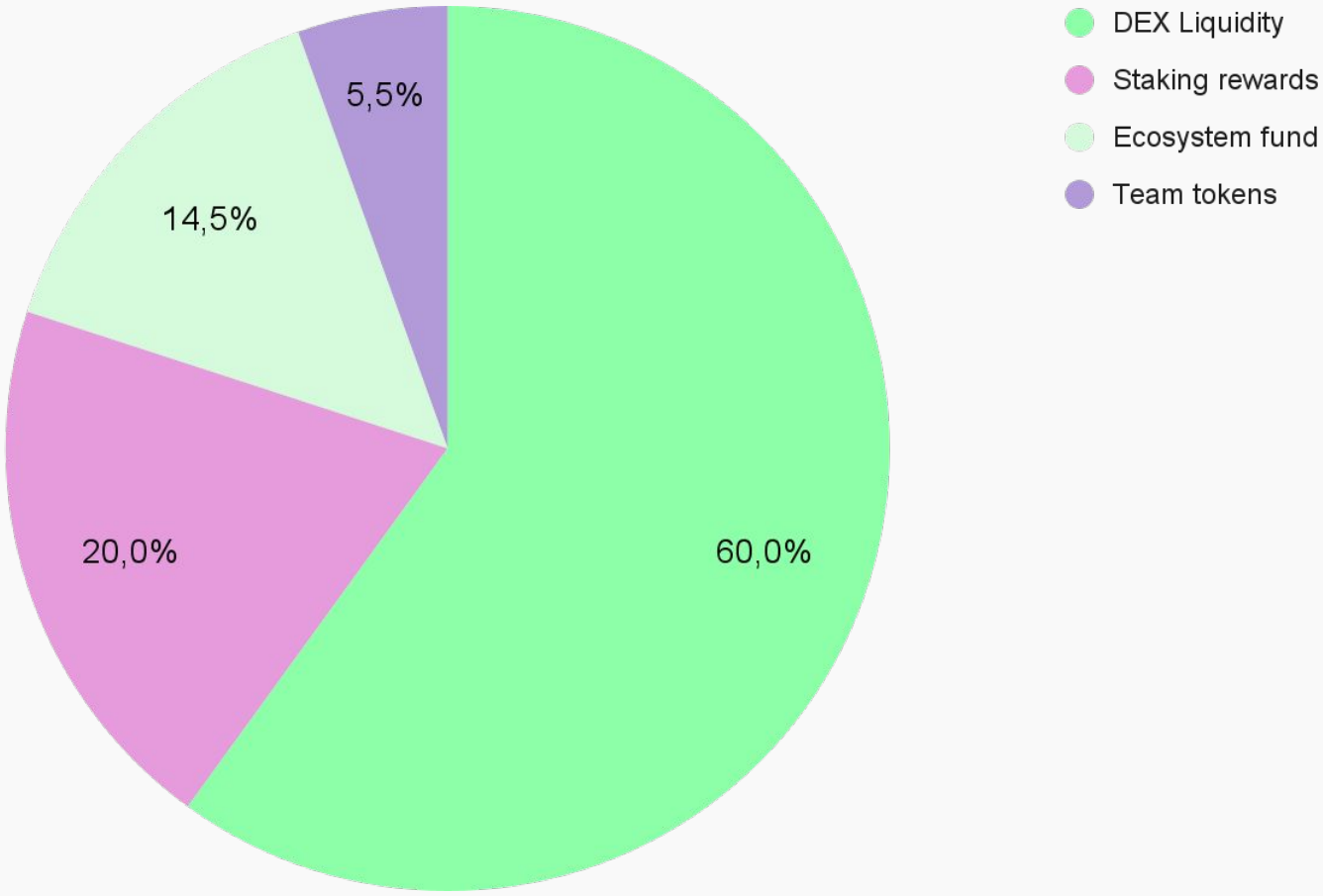
Owner can exclude addresses from taxes.

```
function renounceWhitelist() external onlyOwner {
    _renounced.Whitelist = true;

    emit RenouncedWhitelist();
}

function whitelist(address account, bool status) public onlyOwner {
    _whitelist(account, status);
}

function whitelist(address[] calldata accounts, bool status) external onlyOwner {
    unchecked {
      uint256 cnt = accounts.length;

      for (uint256 i; i < cnt; i++) { _whitelist(accounts[i], status); }
    }
}

function _whitelist(address account, bool status) internal {
    require(!_renounced.Whitelist);
    require(account != address(0) && account != address(0xdEaD));
    require(account != _dex.router && account != _dex.pair, "DEX router and pair are privileged");


    _whitelisted[account] = status;

    emit Whitelisted(account, status);
}
```

TOKENOMICS

The following tokenomics are based on the project's whitepaper and/or website:

- 60% - DEX Liquidity
- 20% - Staking rewards
- 14.5% - Ecosystem fund
- 5.5% - Team tokens



WEBSITE

Website URL
https://chainfactory.app/

Domain Registry
https://domains.google.com

Domain Expiration
2024-05-28

Technical SEO Test
Passed

Security Test
Passed. SSL certificate present

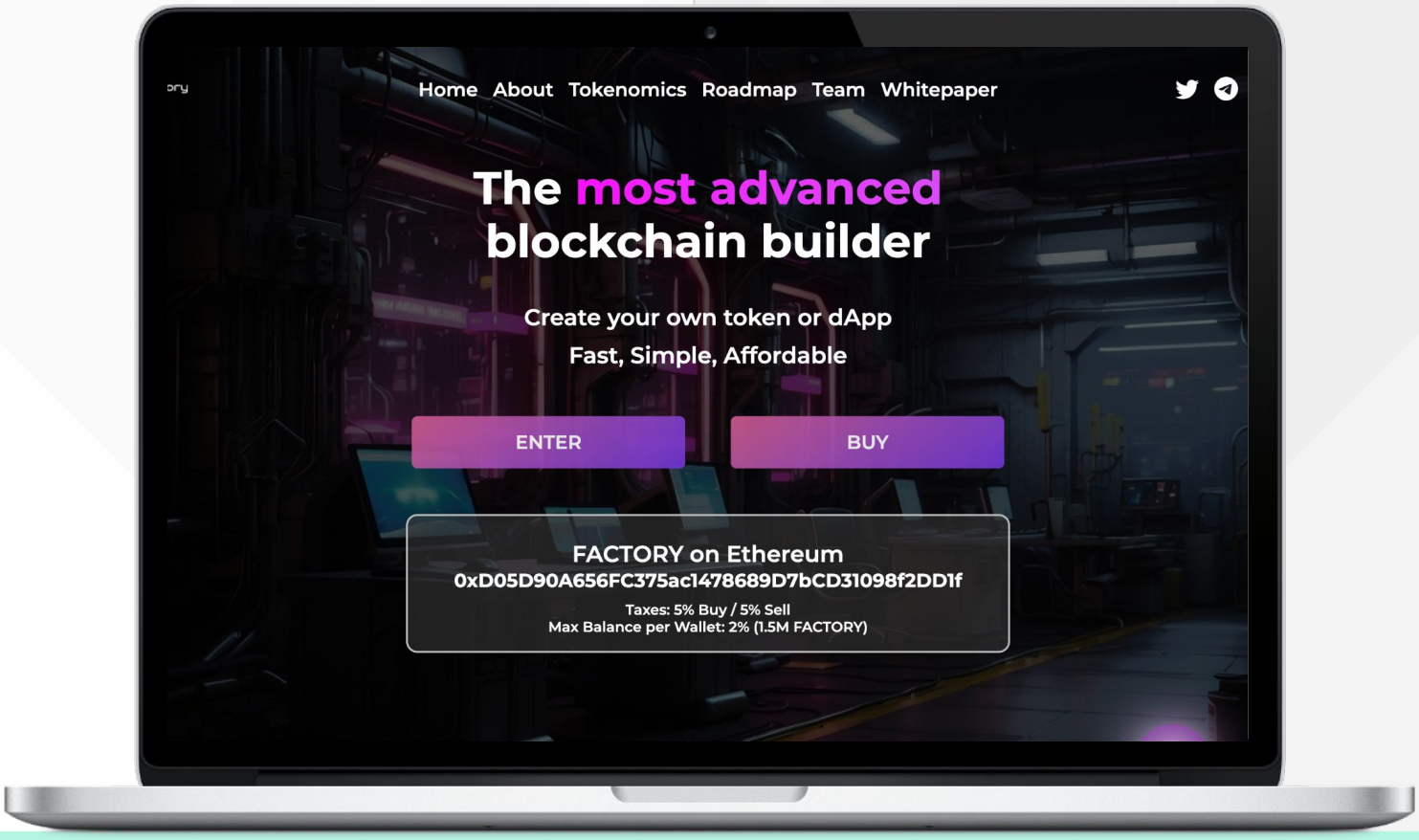
Design
Simple and intuitive web design with appropriate color scheme and graphics.

Content
The information helps new investors understand what the product does right away. No grammar mistakes found.

Whitepaper
Well written, explanatory.

Roadmap
Yes, goals set without time frames.

Mobile-friendly?
Yes



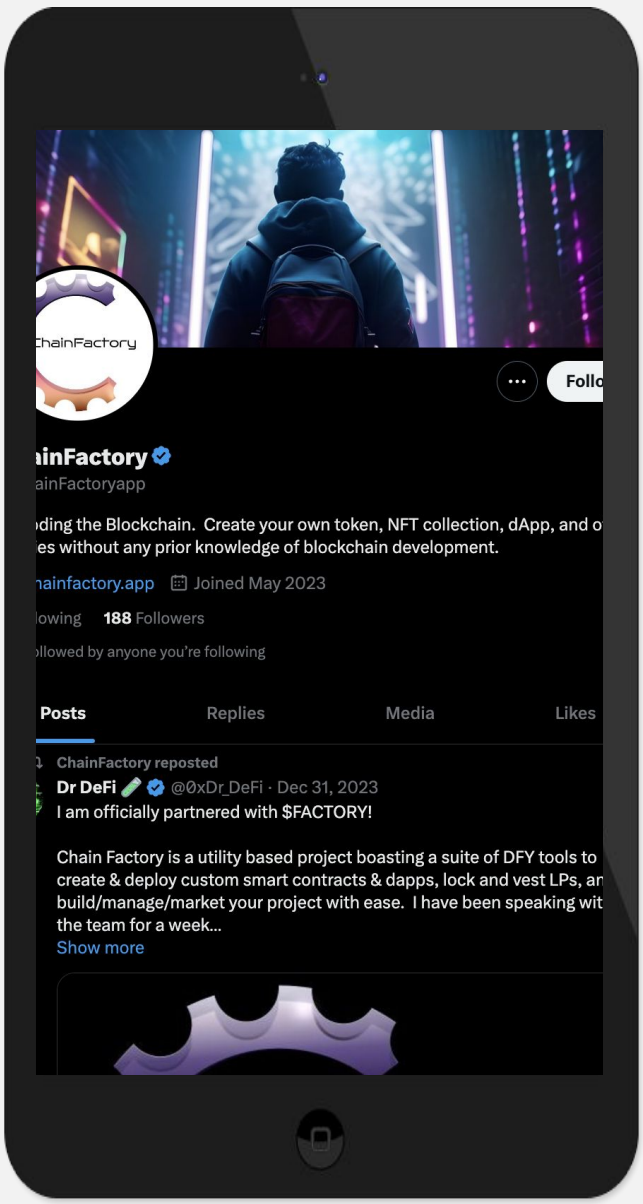
chainfactory.app

SOCIAL MEDIA

& ONLINE PRESENCE

ANALYSIS

Project's social media pages are active.



Twitter's X

@ChainFactoryApp

- 717 followers
- Posts frequently
- Active

Discord

https://discord.com/invite/4eDJf6UwP4

- 140 members
- Active members
- Active mods

Telegram

@ChainFactoryVerify

- 1 297 members
- Active members
- Active mods

Medium

- Not available

SPYWOLF

CRYPTO SECURITY

Audits | KYCs | dApps
Contract Development

ABOUT US

We are a growing crypto security agency offering audits, KYCs and consulting services for some of the top names in the crypto industry.

- **OVER 700 SUCCESSFUL CLIENTS**
- **MORE THAN 1000 SCAMS EXPOSED**
- **MILLIONS SAVED IN POTENTIAL FRAUD**
- **PARTNERSHIPS WITH TOP LAUNCHPADS, INFLUENCERS AND CRYPTO PROJECTS**
- **CONSTANTLY BUILDING TOOLS TO HELP INVESTORS DO BETTER RESEARCH**

To hire us, reach out to
contact@spywolf.co or
t.me/joe_SpyWolf

FIND US ONLINE

SPYWOLF.CO

@SPYWOLFNETWORK

@SPYWOLFNETWORK

Disclaimer

This report shows findings based on our limited project analysis, following good industry practice from the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, overall social media and website presence and team transparency details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report.
While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER:
By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice.
No one shall have any right to rely on the report or its contents, and SpyWolf and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SpyWolf) owe no duty of care towards you or any other person, nor does SpyWolf make any warranty or representation to any person on the accuracy or completeness of the report.
The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SpyWolf hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SpyWolf hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SpyWolf, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts, website, social media and team.
No applications were reviewed for security. No product code has been reviewed.